Advances in autonomy for small UGVs

Michael H. Bruch*, Jason Lum, See Yee, Nghia Tran
SPAWAR Systems Center, San Diego, 53560 Hull St., San Diego, CA 92152

ABSTRACT

Many advances have been made in autonomy for unmanned ground vehicles (UGVs) but most of these advances have been for large UGVs only, in that the sensors required for autonomy are typically large, heavy and require a significant amount of power. Because of the size, weight and power restrictions imposed by a man-portable UGV advances in autonomy have been very limited.

The SPAWAR Systems Center San Diego (SSC San Diego) has previously developed a waypoint navigation capability for small robots. That system required an operator to monitor a live video feed from the vehicle to ensure it did not strike any obstacles in its path. Now SSC San Diego in cooperation with the NASA Jet Propulsion Laboratory (JPL) has developed a miniature obstacle detection sensor suitable for small robots. SSC San Diego has also developed the obstacle-avoidance algorithms to navigate autonomously around obstacles.

Keywords: robotics, unmanned ground vehicle, UGV, obstacle avoidance, OA, autonomous, man-portable

1. INTRODUCTION

In 1999 under the Joint Robotics Program (JRP) Man-Portable Robotic Systems (MPRS) project, SSC San Diego developed a small UGV intended for use by Army engineers for tunnel, sewer, cave, and urban structure reconnaissance. As originally developed, the UGV (called the URBOT for Urban Robot, Fig. 1), was strictly tele-operated from a wearable operator control unit (OCU) (Fig. 2). This allowed the soldier to manually drive the vehicle into high-risk areas and receive video feedback to assess the situation before entering. The system was used in several experiments at Fort Leonard Wood, MO, Fort Drum, NY, and Fort Polk, LA.[1]

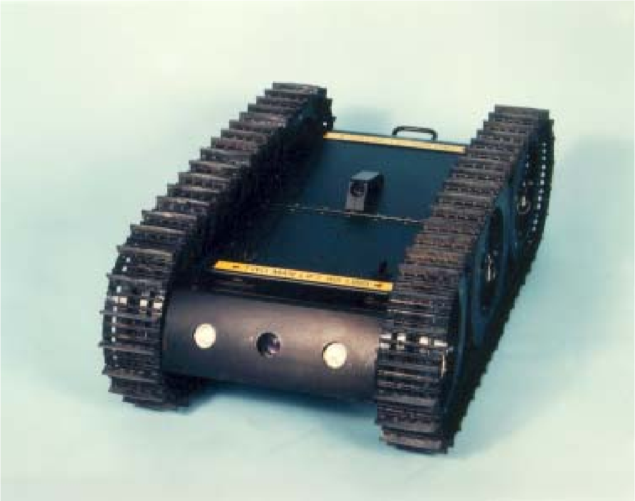


Figure 1. MPRS URBOT

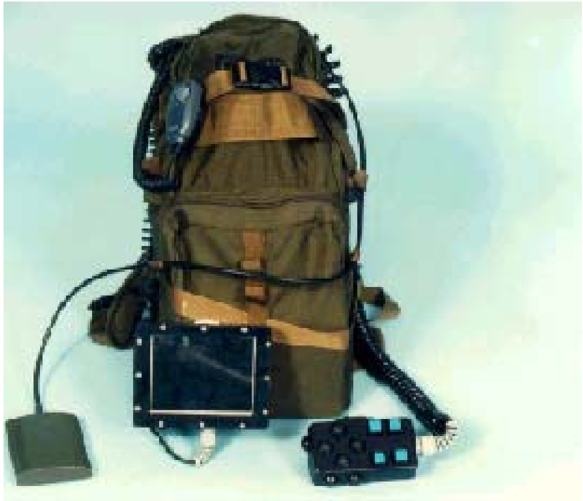
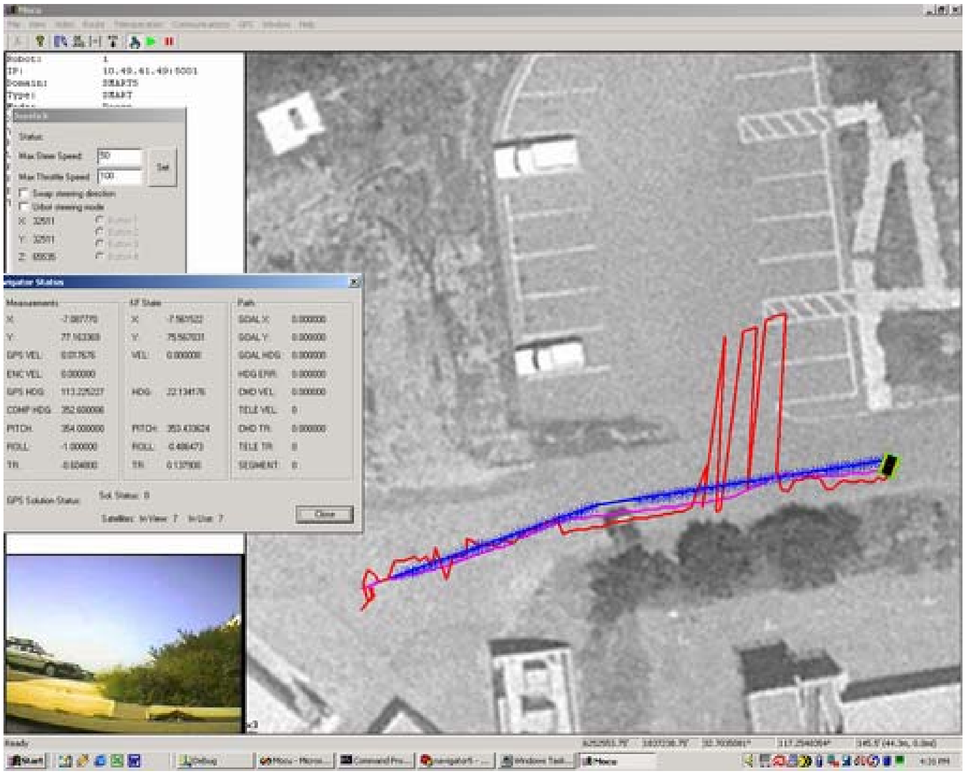


Figure 2. MPRS URBOT OCU

In 2002 SSC San Diego was tasked with developing a GPS-waypoint-navigation capability for the URBOT. The requirement was for a small robot to semi-autonomously navigate undetected several miles into a combat area. The user was concerned about operator fatigue if the system was to be tele-operated the entire distance. The concept of operations was to pre-plan the mission using high resolution imagery, and then to select a path that appeared to

*michael.bruch@navy.mil; phone 1 619 553 7977; fax 1 619 553 5578; www.spawar.navy.mil/robots obstacle-free. The operator would monitor the video feed from the vehicle and watch for obstructions, taking manual control of the vehicle only when necessary to drive around an obstacle. Under this task, SSC San Diego developed one of the first waypoint navigation systems for UGVs of this size[2], as well as the Multi-robot Operator Control Unit (MOCU) which continues to evolve, and is now the core command and control station for the Spartan ACTD. Figure 3 shows a screen shot of an early version of MOCU. The planned route is in blue, the raw GPS data (with large spikes) is red, and the actually URBOT path (determined by the Kalman Filter) is in magenta.


Robot: 1
IP: 10.49.41.49:5001
Domain: SMARTS
Type: SMART

Status:
Max Steer Speed: 50
Max Throttle Speed: 100
□ Swap steering direction
□ Urbot steering mode
X: 32511
Y: 32511
Z: 65535

Measurements		KF State		Path	
X:	-7.087770	X:	-7.567522	GOAL X:	0.000000
Y:	77.163369	Y:	75.567031	GOAL Y:	0.000000
GPS VEL:	0.017676	VEL:	0.000000	GOAL HDG:	0.000000
ENC VEL:	0.000000			HDG ERR:	0.000000
GPS HDG:	113.225227	HDG:	22.134176	CMD VEL:	0.000000
COMP HDG:	352.600006			TELE VEL:	0
PITCH:	354.000000	PITCH:	353.433624	CMD TR:	0.000000
ROLL:	-1.000000	ROLL:	0.486473	TELE TR:	0
TR:	-0.024000	TR:	0.137900	SEGMENT:	0

GPS Solution Status: Sol Status: 0
Satellites: In View: 7 In Use: 7

Figure 3. Screen shot of an early version of MOCU

Autonomous obstacle avoidance (OA) was not attempted for this task primarily because of the lack of adequate sensors that met the size, weight, power and fidelity requirements of a rugged, man-portable system. A great deal of successful work has been done OA for larger UGVs, but those systems rely on sensors that are large, power-hungry, and require CPU intensive operations and are inappropriate for a small robots.

In recent years it has become increasingly clear that small robots provide a needed capability to the warfighter. It is also clear that as small robots become more prevalent the requirement for higher levels of autonomy will increase [3]. In Iraq soldiers are using and depending on small robots daily but only in situations that are controlled and pose a serious risk of injury or death. One reason for this is that all of the small robots currently fielded are strictly tele-operated. These tele-operated systems require the operator's full attention and prevent him from maintaining his own personal security. The primary example of this is in Improvised Explosive Device (IED) disposal. In this application the risk to the soldier in manually disposing of the IED is huge. In most cases IED disposal is also done in a relatively controlled environment where the robot operator is in an area secured by a force protection team.

In order for small robots to become more useful in more tactical scenarios, they must have autonomous navigation capabilities. This is also the case for applications that would make use of teams of small robots, such as land mine clearance, communications relaying, etc. One of the most basic building blocks for any autonomous behavior is robust obstacle avoidance.

The requirement for developing sensors and autonomous capabilities for small platforms is also influenced by the platforms themselves. The primary small robot acquisition programs currently underway within the DoD are the Future Combat Systems (FCS) Soldier Unmanned Ground Vehicle (SUGV) and the NAVEOD Man-Transportable Robotic System (MTRS). The FCS SUGV platform will be approximately 25% smaller than the iRobot Packbot. In order for that vehicle to maintain its mobility (including self-righting) and survivability, any sensor system must be integrated into a very small payload area between the tracks. This will require the sensors to be extremely small by today's standards and virtually eliminates most of the sensors currently being used in academic and scientific research.

2. PREVIOUS WORK IN AUTONOMY FOR SMALL UNMANNED GROUND VEHICLES

There is a great deal of research being conducted throughout the academic and scientific community on robotic autonomy and intelligence that use small robot platforms for development and testing. The vast majority of these efforts are using relatively inexpensive research platforms such as the ActivMedia Pioneer or iRobot ATRV-mini, which are not designed for outdoor military type applications. In fact, much of the research conducted using small robots has been focused primarily on indoor applications. Furthermore, the majority of this work has addressed algorithms and behaviors and has made use of sensors that provide the most accurate and complete data. As a result most of these projects have developed highly sophisticated behaviors that require sensors and processors that are generally not suitable for integration on a small, field-ready robot that must be environmentally sealed. The following discussion addresses past and present research that provides the best opportunity for transition to a small, rugged, outdoor robot.

Much of the early relevant work in autonomy for small robots was done under the DARPA Tactical Mobile Robot (TMR) program from 1998 to 2002. The TMR program had several goals, one of which was autonomy in urban environments, where the objective was to demonstrate robust traversal of complex terrain with minimal human intervention (less than 1 command per 50 meters). Under the TMR program, NASA's Jet Propulsion Laboratory (JPL) demonstrated OA using stereo vision.[4] During development and testing, JPL found that noise in the infrared (IR) and sonar sensor data precluded their use as part of the OA system. This matches what SSC San Diego has found through various attempts at using small IR and sonar sensors for OA in outdoor environments.

The JPL stereo cameras used for the TMR program had a field of view (FOV) of 97x47 degrees. The stereo imagery was processed into 80x60-pixel disparity maps using a 233-MHz Pentium II PC/104 processor stack. An obstacle-detection algorithm[5] was directly applied to the disparity map, which resulted in three distinct classifications: no obstacle, traversable obstacle, or nontraversable obstacle. This data was projected onto a 2-D occupancy-grid map that was 2.5x2.5m with cells of 10cm. The OA algorithm used by JPL was an adaptation of the Carnegie Mellon University (CMU) Morphin algorithm.[6] This algorithm evaluates a predetermined set of steering arcs that pass through the occupancy grid, penalizing arcs that are blocked by nontraversable obstacles or that pass through cells with traversable obstacles. The votes generated by this algorithm for each arc are then passed to the arbiter for combination with arc votes from other behaviors, such as waypoint following. This is the basis for work that SSC San Diego is conducting for the JRP MPRS project.

Currently, iRobot Corporation is investigating autonomy on their Packbot man-portable robot. This work is in support of the Tank-Automotive Research, Development and Engineering Center (TARDEC) Wayfarer and Sentinel programs. The Wayfarer program is focused on autonomous urban reconnaissance using small UGVs. Objective capabilities include waypoint navigation, obstacle avoidance, autonomous street following, building mapping, video and data recording and more. For sensors iRobot is primarily using a SICK LDA 360-degree laser rangefinder, a Point Grey Bumblebee stereo vision system, an Indigo Omega FLIR and the organic Packbot GPS receiver. This a very ambitious program that has shown promising results, but still does not address the sensor to robot size, weight and power ratio problems. These current sensors (especially the laser) will not fit on a SUGV-sized robot.

3. **SENSOR DEVELOPMENT**

Over a period of two years SSC San Diego has worked with JPL to develop a miniaturized stereo vision system that could be used for OA on a small robot. The effort leveraged JPL's experience with stereo vision systems from the DARPA TMR program, Mars Rover programs, and resources from the DARPA Micro Air Vehicle program. The goal of this collaboration was to develop a small low-power obstacle-detection sensor for small robots.

The result of this collaboration and combined funding was a small camera board called the JPL SmartCam (Fig 4). This board includes both the processing and imaging elements so that no external hardware (imagers or processors) is required. The board measures 2 inches square and holds a Kodak 640x480 imager, a Xilinx Vertex11 FPGA, a Motorola Starcore DSP, 32MB of SDRAM and 4MB of flash memory. Inter-board communications (for stereo vision processing) are currently done via a high-speed serial bus between the DPS cores. There are also options for directly communicating between FPGAs but they are not being used at this time. The FPGA interfaces directly to the Kodak camera to collect imagery and performs some of the image preprocessing functions. The image data is then transferred to the DSP via the memory bus where the final stereo vision processing is performed. The FPGA is also used to output an NTSC signal that displays raw image data as well as various types of processed data.

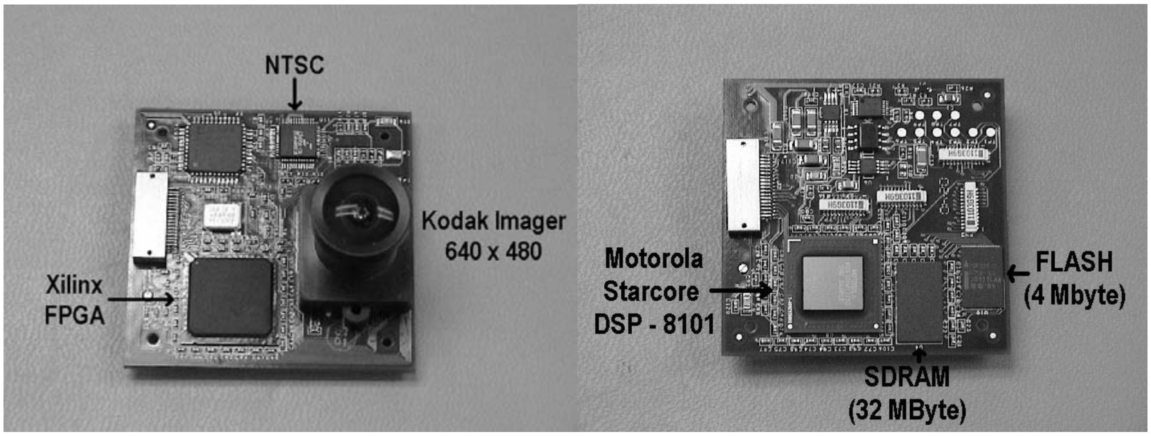


Figure 4. JPL Smart Camera, front and back

The DSP is running a scaled-down version of the JPL stereo vision software, which generates 160x120 pixel disparity maps. Figure 5 shows an example image and the corresponding disparity map, shown in grayscale, where the brighter parts of the image are closer to the camera. From the disparity map an obstacle detection algorithm generates a two-dimensional occupancy grid-like obstacle map. This obstacle map is output over a TTL-level serial port for use by the obstacle-avoidance system. The obstacle map is a 7x7m grid with 41 cells along each side, resulting in 1681 cells that are approximately 17cm square. With the current software/firmware implementation the obstacle map is output at a rate slightly under 2Hz.

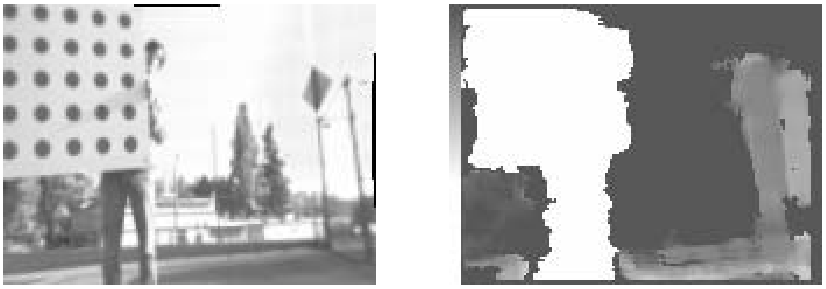


Figure 5. Left image is rectified image from left stereo camera; right image is the corresponding disparity map

The obstacle-detection algorithm effectively looks at the slope of the terrain to determine where obstacles lie. Consider Figure 6, which shows an example terrain profile with points X and Xa and the corresponding pixels in the image (P and Pa). The algorithm scans through each pixel P in the image and the associated 3-dimensional point X in space. It then calculates a ray which passes through the focal point F and a point (Y) located above X and equal to the minimum obstacle height value. The corresponding pixel for that point (Pa) is found and used to calculate its associated 3D point Xa. The pixel P is then considered to be an obstacle if the distance from the camera to Xa is shorter than to X or if the slope from X to Xa is greater than a predetermined threshold. The cell in the 2D obstacle map that corresponds to X is marked as an obstacle if the threshold value of pixels-per-obstacle is met or exceeded for that cell. Requiring obstacles to exceed the pixels-per-obstacle threshold helps filter out noisy data.

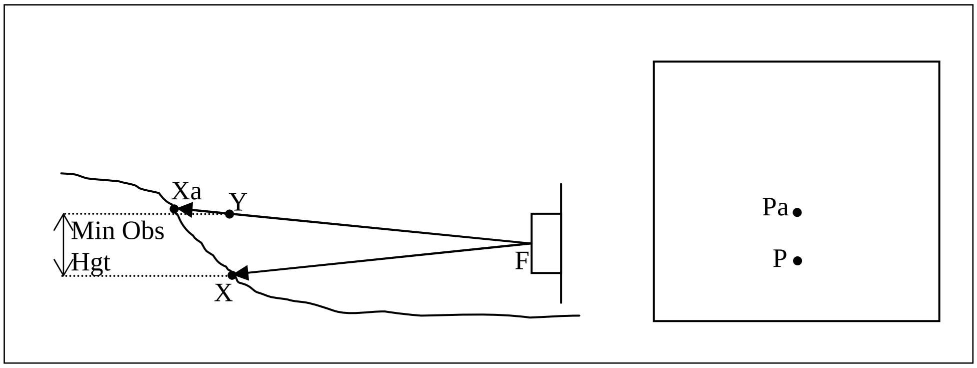


Figure 6. Depiction of obstacle detection

SSC San Diego has conducted limited testing with the Smart Camera stereo vision sensors prior to installation in the robot, with the sensors at approximately the same height from the ground as they will be on the robot (22cm). The tests show that in relatively flat open terrain the cameras produce good results, successfully identifying obstacles ranging from large curbs to tumble weeds to steep slopes. Figure 7 shows an example image and the corresponding obstacle map.

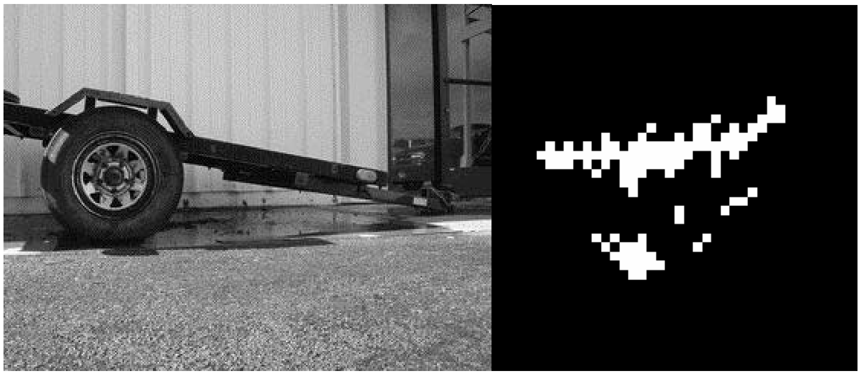


Figure 7. Image from stereo camera and corresponding obstacle map

More thorough tests are planned for the Smart Camera stereo vision system when installation on the URBOT is complete. To date the mechanical integration is complete and the calibration to compensate for the camera alignment in the new housing is in progress.

4. OBSTACLE AVOIDANCE

The obstacle-avoidance algorithm developed for the URBOT is loosely based on the CMU Morphin algorithm, a behavior-based navigation architecture where various behaviors vote on a discrete set of actions.[6] As applied here a number of arcs are projected in front of the vehicle over the obstacle map (Fig. 8). The number of arcs considered is a function of the map size and grid spacing with the arcs spaced such that one arc passes through each of the outer cells. This guarantees that each cell in the grid is covered by at least one arc. For a 7x7m grid with 41x41 cells, this results in a total of 160 arcs, 80 on each side of center. The arcs to the left of center are considered positive because a left turn has a positive turn rate (right-hand rule with Z up). Each of those arcs is related to the vehicle velocity and turn rate by

$$R = \frac{V}{\dot{\theta}}$$

where R is the radius of the arc, V is the vehicle velocity and $\dot{\theta}$ is the vehicle turn rate.

Each arc is given a weight or vote based on the distance the robot could travel along that arc before it encountered an obstacle. The longer arcs are weighted more favorably than the shorter arcs or arcs that are blocked near the robot. The votes are scaled from 1 to -1 so that they can be combined with votes from other behaviors by the arbiter (Fig. 10, top right).

Prior to running the OA routines the obstacles in the OA are expanded or grown. Growing obstacles is a common method used to compensate for the simplification of describing the robot as a point rather than a polygon. The obstacles are generally grown by half the length of the longest dimension of the vehicle, assuming that the vehicle's coordinate system origin is in the center of the vehicle.

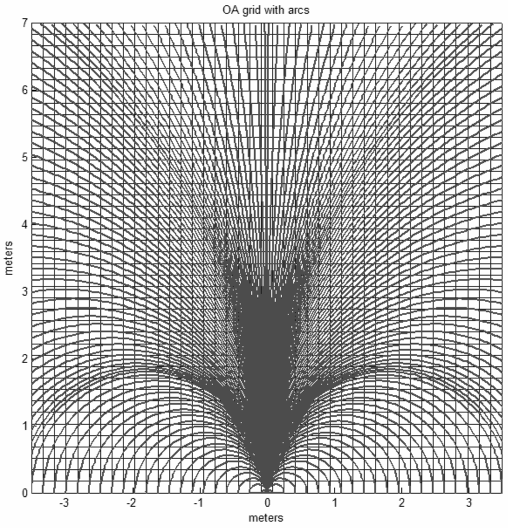


Figure 8. Obstacle grid with arcs shown

The votes for each arc from the OA algorithm are passed to the navigation arbiter where they are combined with votes from other navigation behaviors. The primary navigation behavior is waypoint navigation or route following. The waypoint navigation function produces a commanded turn rate and velocity based on the heading error between the robot's current heading and the heading toward the goal point along the path (Fig. 9). At each cycle these turn-rate and velocity commands from the waypoint navigation function are converted into an arc. To form votes for the entire array of arcs, the primary arc is given the maximum value and votes for arcs on either side of it are linearly decreased until they reach a value of zero (the path following behavior does not vote against any arcs) (Fig. 10, top left).

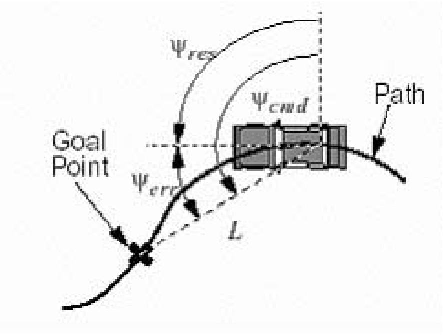


Figure 9. Depiction of heading error between robot's heading and heading to the goal point (Kelly, 1997)

Other arc-voting behaviors have also been added to help navigate around obstacles. These include a free-space behavior that votes (from 1 to 0) for large continuous sections of arcs that are unblocked, essentially open areas in the obstacle map (Fig. 10, bottom left). This behavior also votes for open arcs that fall between obstacles so that the robot won't always favor going around groups of obstacles. An additional behavior was added that favors the arcs with the greatest radius, or in other words favors the current vehicle heading, which helps prevent sudden changes or oscillations in the vehicle heading.

Weighting factors are applied to the votes from each of the behaviors prior to combining them so that some behaviors contribute more to the final outcome than others. In the current implementation the OA behavior is weighted almost three times heavier than any of the other behaviors. The combined votes are shown in the bottom right plot in Figure 10.

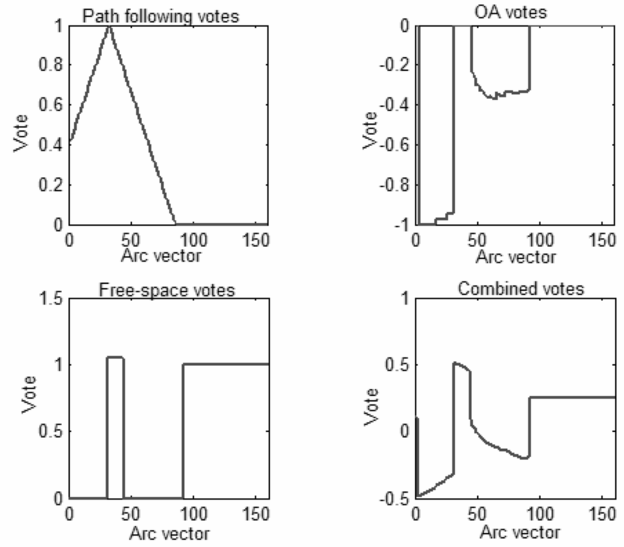


Figure 10. Example votes from behaviors

5. SIMULATION

To develop and test the OA algorithm a simulation was developed in Matlab, basically an extension of the simulation that was used to develop the original Kalman Filter and waypoint navigation functions. Based on the success of transferring the original simulation algorithms to the real robot it is expected that the obstacle-avoidance simulation will transfer successfully as well.

The OA simulation allows the user to define the size of the simulation area, draw a route to follow, set the desired vehicle speed, pick the robot's initial position and orientation, pick locations for obstacles, and set the size of the obstacles. When the simulation is run, the path-following routine "drives" the robot along the route. The arbiter combines the commanded velocity and turn rate output from the path following behavior with the arc votes from the OA behaviors as necessary to avoid the simulated obstacles. An example of a simulated run is shown in Figure 11, showing how the robot successfully avoided the obstacles while following the planned route.

The simulation has proven to be invaluable. It has allowed multiple test runs under varying conditions to be conducted with different behaviors over a short period of time. Because there a several degrees of randomness built into the vehicle simulation, the test runs never produce identical results and more closely approximate the real robot. One of the most important results of the simulation is to illustrate how the relatively narrow field-of-view (FOV) of the stereo sensors (~90°) coupled with the relatively slow update rate places significant restrictions on the vehicle dynamics. To compensate for the narrow FOV and slow update rate, the simulation was modified such that the data in the obstacle map is dead reckoned using the state information from the robot. The primary benefit of this is that the obstacles are held in the obstacle map even after they leave the FOV of the cameras. This is essential to prevent the OA algorithm from turning the robot into an obstacle that the robot is passing. This technique has proven useful in simulation but it is not clear yet how well the dead reckoning will work on the actual robot. Additional behaviors have been developed that limit the turn rate to ensure the vehicle does not turn greater than the FOV of the cameras between obstacle map updates, but they have not been tested.

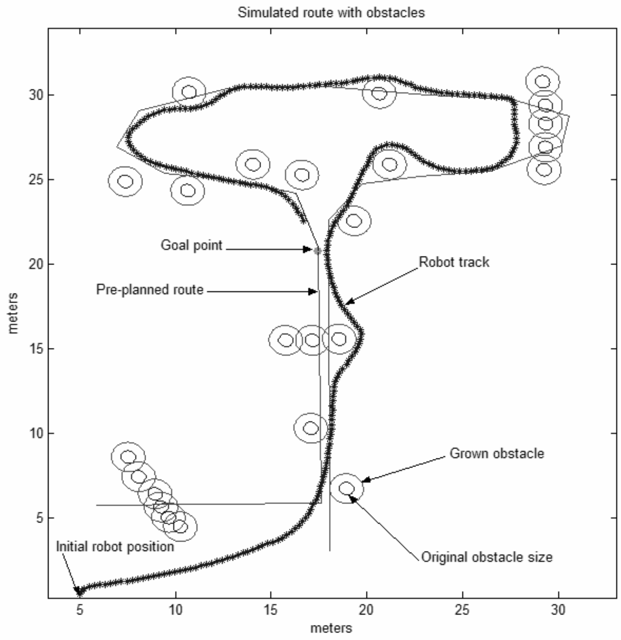


Figure 11. Example of OA simulation run

6. VEHICLE IMPLEMENTATION

A new modular sensor suite was built to house the Smart Cameras on the URBOT (Fig 12). The stereo cameras are fixed on a 10cm baseline which is easily accommodated between the URBOT tracks. To maintain the inspection and surveillance capability of the URBOT, a small Sony block camera is mounted between the stereo pair. On the outside of either Smart Camera a halogen head light is mounted. These lights are primarily for tele-operation at night but will also be used to test the feasibility of using stereo vision under vehicle illumination.

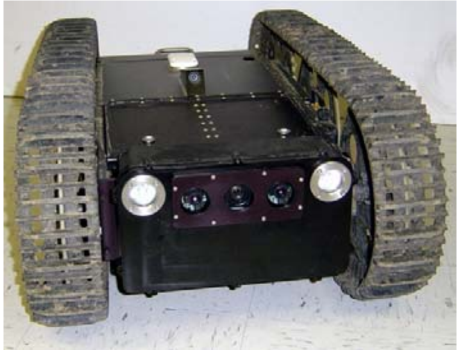


Figure 12. New URBOT sensor housing with stereo vision cameras

The sensor housing also holds the Observer CPU, which is responsible for controlling all the hardware in the housing, collecting data from the stereo cameras, processing the data and sending the arc votes to the navigation processor. The Observer CPU is a CM-i686 single-board computer (SBC) from Compulab. The processor is a National Semiconductor Geode clocked at 300MHz and is running a Linux operating system. The Compulab SBC is stacked on top of a breakout board developed at SSC San Diego (Fig. 13). The breakout board (called the 686 daughter board) is used to bring out the connections of the 686 to connectors suitable to use for integration with other systems. The daughter board also adds significant capabilities, including a 1-million-gate FPGA and an 8051 microcontroller with CAN bus and digital to analog outputs. The daughter board can also be used in a standalone mode without the 686 SBC. Combined, the 686 and daughter board provide 10/100 Ethernet, CAN bus, 7 serial ports, 3 host USB ports, an I2C bus, A/D, D/A, over 50 digital I/O lines and more. The daughter board has a footprint of 7.1x11.2cm and together with the 686 SBC requires only about 4W.

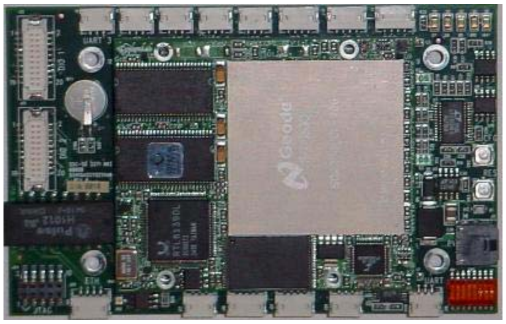


Figure 13. SSC San Diego 686 daughter board with Compulab CM-i686 SBC

The obstacle-avoidance software has been ported from the Matlab simulation to the 686 processor. Using simulated data for the obstacle map, the CPU is able to process the data (including dead reckoning the map) and output the arc votes at a 10-Hz rate using less the 5% of the CPU capacity.

7. CONCLUSIONS

The results of this effort to date have yielded several conclusions. The first is that the miniature stereo vision system using the JPL Smart Cameras is suitable for robust integration into a small UGV. The system is very small, light-weight and requires relatively little power. It should be feasible to integrate these sensors into an FCS-SUGV-size vehicle without degrading the vehicle’s mobility, survivability or even limit the available payload space to a significant degree. The initial sensor data collected at SSC San Diego is very promising. The sensors are able to function effectively outdoors in varying lighting conditions, and the output data appears to be sufficiently noise-free with few false obstacles. The ability to accurately detect a wide range of obstacles of different sizes and textures is also promising.

The current drawbacks of the Smart Camera sensors include the relatively slow update rate and the low-grade optics. The slow update rate is a function of processing power. Currently, most of the stereo image processing is done in the DSP. It may be possible to make more use of the FPGA to share the processing load with the DSP. It is also likely that the software could be further optimized. The optics were chosen based on the size requirements for the Smart Camera. The resolution is adequate for this application at 640x480 but the poor quality of the lenses and difficulty aligning the lens housing with the imager during fabrication results in less than optimal performance.

Varying versions of the OA algorithm employed here have been used on many successful robotic vehicles including the NASA Mars Rovers, the CMU NavLab and others. It has proven to be a very flexible architecture that is easily adapted to different vehicles for different requirements. The algorithm used on the SSC San Diego URBOT is a fairly crude instantiation but also allows real-time performance on relatively low-power processors. The simulation is working well but the real-world effectiveness of this work will not be known until it can be tested on the vehicle.

8. REFERENCES

1. M. Bruch, R. Laird, H. Everett, *Challenges for Deploying Man-portable robots into hostile environments*, SPIE Proc. 4195: Mobile Robots XV, Boston, MA, November 2000
2. M. Bruch, G. Gilbreath, J. Muelhauser, *Accurate waypoint navigation using non-differential GPS,* AUVSI Unmanned Systems 2002, Lake Buena Vista, FL, July 2002
3. Pacis, E.B., Everett, H., Farrington, N., Sights, B., Kramer, T., Thompson, M., Bruemmer, and D., Few, D., "Transitioning Unmanned Ground Vehicle Research Technologies," SPIE Proc. 5804: Unmanned Ground Vehicle Technology VII, Orlando, FL, March 29 - 31, 2005.
4. L. Matthies, Y. Xiong, R. Hogg, *A Portable, Autonomous, Urban Reconnaissance Robot*, Intelligent Autonomous Systems, Venice, Italy, July 2000.
5. L. Matthies, A. Kelly, T. Litwin, G. Tharp, *Obstacle detection for unmanned ground vehicles: a progress report*, Robotics Research: The Seventh International Symposium, Springer-Verlag, 1996.
6. R. Simmons, L. Henriksen, L. Chrisman, G. Whelan, *Obstacle avoidance and safeguarding a lunar rover*, Proceedings from the AIAA Forum on Advanced Developments in Space Robotics, Madison, WI, August 1998